IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Instalación MongoDB mediante docker

Siguiendo las instrucciones de la web oficial de Docker vamos a instalar docker desde su repositorio oficial. De esta manera tendremos acceso a las actualizaciones oficiales tan pronto como se publiquen.

Instalación de Docker.

En versiones actuales de Ubuntu Server, incluso con una instalación mínima ya deberíamos tener instalados algunos paquetes que serán necesarios como ca-certificates o gnupg

```
sudo apt install ca-certificates gnupg
```

Lo primero que necesitaremos es una clave pública para poder acceder al repositorio de docker. La descargaremos directamente desde la web de Docker mediante el siguiente comando.

```
wget https://download.docker.com/linux/ubuntu/gpg
```

Con este comando nos hemos descargado la llave pública de Docker en versión texto plano.

```
sudo gpg --dearmor -o /usr/share/keyrings/docker.gpg gpg
```

Con este comando descompactamos y pasamos a binario la llave pública en formato texto para que apt pueda utilizarla. Ahora ya podemos crear un nuevo repositorio para apt.

```
sudo nano /etc/apt/sources.list.d/docker.list
```

Con este comando creamos un nuevo fichero con extensión list y que servirá a apt para añadir un repositorio desde donde descargar Docker. En este archivo añadimos la siguiente línea:

```
deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/docker.gpg ] https://download.docker.com/linux/ubuntu
jammy stable
```

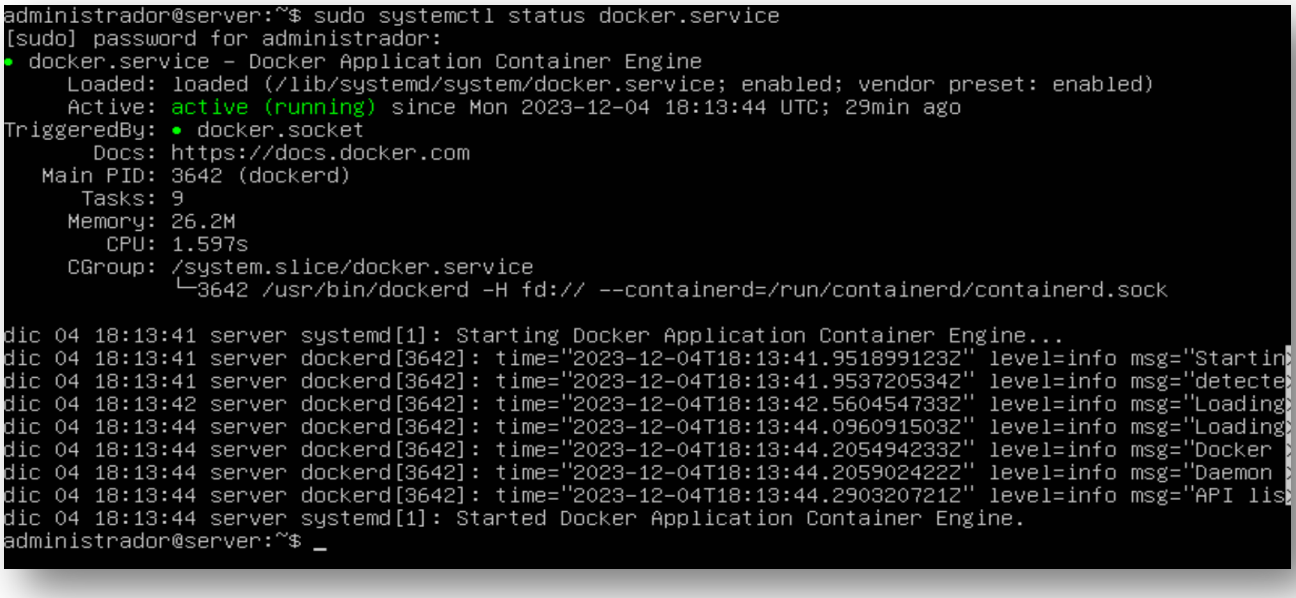
```
administrador@server:~$ sudo gpg -o /usr/share/keyrings/docker.gpg --dearmor gpg
administrador@server:~$ sudo apt update
Obj:1 http://es.archive.ubuntu.com/ubuntu jammy InRelease
Obj:2 http://es.archive.ubuntu.com/ubuntu jammy-updates InRelease
Des:3 https://download.docker.com/linux/ubuntu jammy InRelease [48,8 kB]
Obj:4 http://es.archive.ubuntu.com/ubuntu jammy-backports InRelease
Obj:5 http://es.archive.ubuntu.com/ubuntu jammy-security InRelease
Des:6 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [22,9 kB]
Descargados 71,7 kB en 1s (61,1 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Se pueden actualizar 78 paquetes. Ejecute «apt list --upgradable» para verlos.
administrador@server:~$ _
```

```
sudo apt update
```

Con este comando actualizamos los repositorios. De esta manera incorporamos los nuevos paquetes que hemos añadido a través de repositorio de Docker.

```
sudo apt install docker-ce
```

Tras la instalación de docker se habrá creado también un servicio llamado docker.service que hace que se inicie de manera automática en cada arranque.




```
sudo systemctl status docker.service
```

Si nos fijamos en la salida vemos que el servicio está en ejecución y que está en modo enabled lo que quiere decir que se iniciará automáticamente en cada arranque.

Durante la instalación también se ha creado un usuario y un grupo llamado docker que será el encargado de lanzar docker a un nivel de sistema operativo.

```
sudo usermod -aG docker administrador
```

Con este comando añadimos al usuario "administrador" dentro del grupo de docker haciendo que, de esta manera, también él pueda lanzar y ejecutar docker. Será necesario un reinicio de la sesión para que el usuario administrador quede incorporado definitivamente.

En principio ya tenemos instalado docker. Para probarlo podemos ejecutar una imagen preinstalada de prueba.

```
docker run hello-world
```

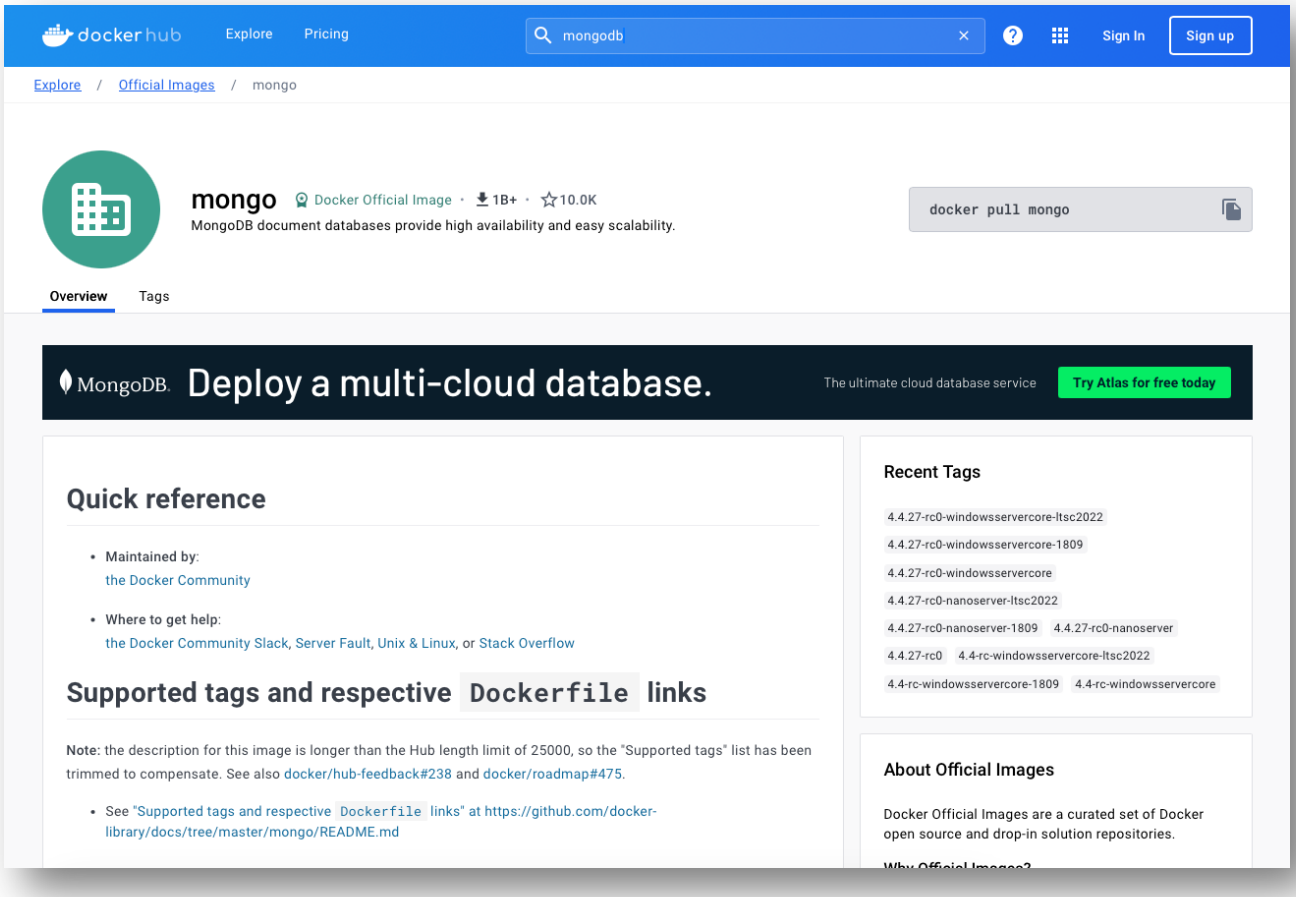
```
administrador@server:~$ sudo docker run hello-world
[sudo] password for administrador:
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:c79d06dfdfd3d3eb04cafd0dc2bacab0992ebc243e083cabe208bac4dd7759e0
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

administrador@server:~$
```

Con este comando descargamos y ejecutamos una imagen muy ligera de prueba llamada hola mundo que lo único que hace es mostrar este mensaje de texto. Con esto es suficiente para saber que tenemos funcionando docker.

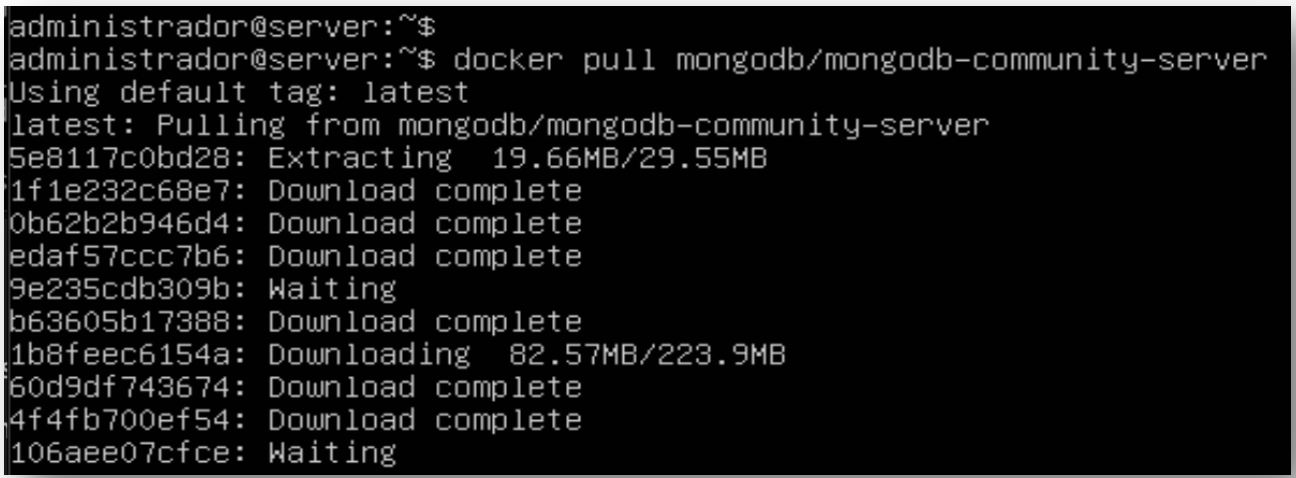
MongoDB en un contenedor.

El siguiente paso consiste en descargarse una imagen de docker de MongoDB. y podemos hacer de, al menos, dos maneras distintas, O bien descargando la imagen desde la web de dockerhub (https://hub.docker.com/) o bien mediante un archivo de texto Dockerfile. También podríamos usar Docker Compose, una variante de Dockerfile en caso de que necesitemos levantar más de un contenedor.

Veremos el primer caso. Si queremos descargar directamente la imagen de la web de dockerhub, será necesario navegar por esa web y buscar la imagen de MongoDB que más nos interese. Debemos tener en cuenta que encontraremos la imagen oficial de MongoDB pero también otras imágenes configuradas por otros usuarios con distintas adaptaciones y usos. Por ello es bueno leer la documentación de la imagen antes de descargar para saber exactamente como está configurada la imagen.



Una vez decida la imagen que queremos usar procedemos a descargarla con el siguiente comando:




```
docker pull mongodb/mongodb-community-server
```

Con este comando descargamos la imagen oficial de MongoDB desde el repositorio de docker.

```
docker run --name mongo -d mongodb/mongodb-community-server:latest
```

Con este comando, al usar el parámetro "run" ejecutamos la imagen "mongodb/mongodb-community-server" que descargamos previamente, opcionalmente se puede especificar la versión en caso de tener varias, aquí lo hacemos con dos puntos y la etiqueta "latest" indicando que queremos ejecutar la última versión disponible.

Además, con el uso del parámetro -d (detach) indicamos que queremos que esa ejecución se realice en segundo plano.

Con el parámetro --name le damos nombre al contenedor donde se ejecutará la instancia de la imagen.

El resultado de ejecutar este comando será una larga secuencia de caracteres que servirá, además del nombre que le hemos puesto, como referencia a este nuevo contenedor.

```
administrador@ubuntuserver:~$ docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS          PORTS       NAMES
2f8b8a045bb5   mongodb/mongodb-community-server   "python3 /usr/local/…"   52 minutes ago   Up 52 minutes   27017/tcp   mongo
administrador@ubuntuserver:~$
```

```
docker ps -a
```

Con el comando anterior podemos listar los contenedores que se han lanzado últimamente. En este caso vemos como el nuestro está en ejecución. En su línea podemos ver la versión corta de la referencia que se le asignó en el momento de su creación. También podemos ver los puertos que tenemos redirigidos y nombre que le hemos dado nosotros, entre otros campos interesantes.

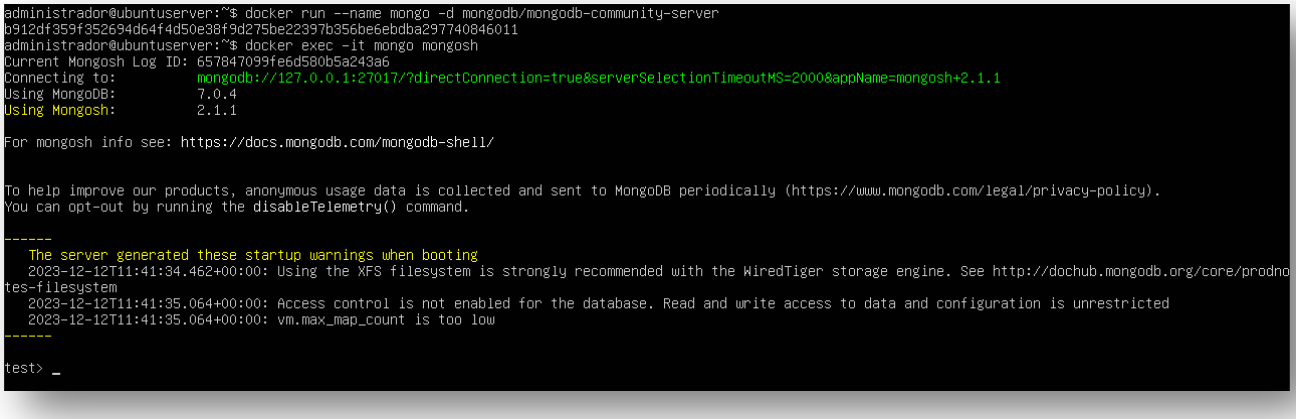
```
docker exec -it mongo mongosh
```

Con este comando ejecutamos el comando mongosh en el contenedor mongo. Recordamos que mongosh es el cliente de MongoDB por interfaz de comandos. Con los parámetros modificamos la ejecución de ese comando creando una terminal con -t y con -i hacemos una sesión interactiva asociando la entrada estándar de nuestro equipo, el teclado.

A efectos prácticos lo que tenemos aquí es una terminal ssh solo que no nos hemos conectado a través del puerto 22 a ninguna máquina remota. Lo que hemos hecho es ejecutar mongosh dentro del contenedor y quedarnos ahí de manera interactiva.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es



Como se puede ver en la captura ya estamos conectados a la interfaz CLI de mongosh y desde esa terminal podemos interactuar con la base de datos.

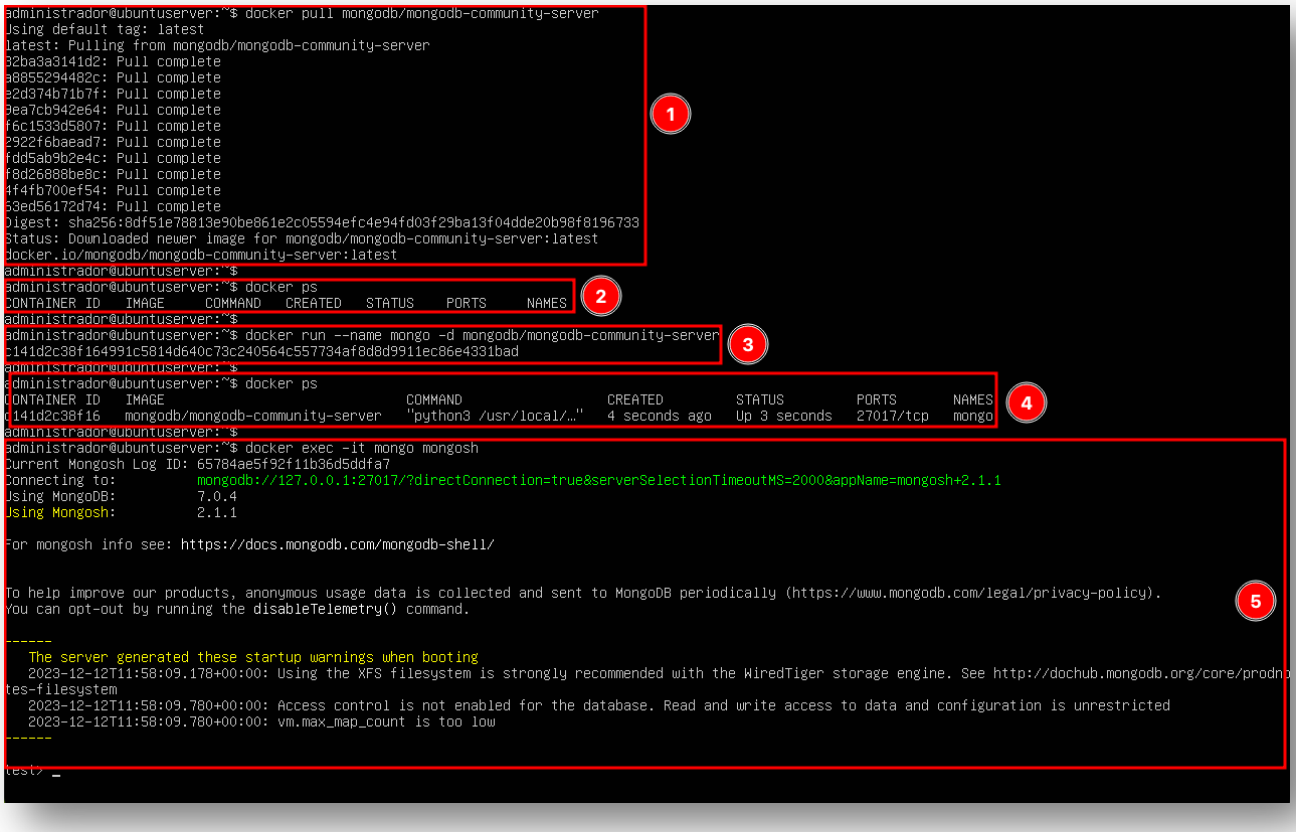
Conclusiones.

Los contenedores son una forma estupenda de virtualizar servicios sin necesidad de crear una máquina virtual clásica donde tendríamos que instalar y configurar desde cero sistema operativo y servicio que necesitamos.

Docker proporciona acceso al kernel de Linux del sistema anfitrión a todos sus contenedores haciendo que no sea necesario instalar uno en cada contenedor.

Las imágenes que nos podemos descargar con un pull desde dockerhub son el conjunto de archivos que trabajan en las capas superiores al kernel. En general las imágenes son compuestas por otras. Por ejemplo, en muchas imágenes se usa una imagen de Alpine que es una distribución Linux basada en debian pero extremadamente ligera, apenas 5 megas y sobre ella se añade el propio servicio como un servidor Web, de bases de datos o cualquier otro.

Con Dockers, además de la eficiencia en el uso de los recursos también contamos con la ventaja de la rapidez con la que se montan los servicios. En la siguiente captura tienes todo el proceso de descarga de una imagen, creación de un contenedor y ejecución de un comando dentro del contenedor.



En la captura anterior podemos ver todo el proceso de descarga de una imagen, creación de un contenedor y ejecución de un comando en el interior del contenedor.

En el paso 1 descargamos la imagen que hemos seleccionado. La imagen por si misma no hace nada, es necesario instanciarla en un contenedor. En el paso 2 mostramos que no hay ningún contenedor en ejecución. Es en el paso 3 donde instanciamos esa imagen en un contenedor. En el paso 4 repetimos el comando para confirmar que, efectivamente, ya tenemos un contenedor en ejecución. Por último, en el paso 5 ejecutamos en comando mongosh dentro del contenedor.

Como podemos ver, poner en marcha un servidor MongoDB lleva solo unos segundos y eso es toda una ventaja.

En la parte de los contras tenemos que se pierde el control sobre la configuración del servicio porque simplemente ya funciona.

```
docker exec -it mongo bash
```

Con este comando abrimos una terminal dentro del contenedor. Con esta terminal podemos ver el archivo de configuración dentro de la carpeta /etc (tiene un nombre ligeramente distinto pero es fácil de localizar), también podremos ver el estado del servicio y, en general, todo lo visto en las guías anteriores.